TP 5

Mise en pratique Maven

Objectifs

Le propos de cet atelier est de se familiariser avec l'outil de construction de build Maven .

Maven : Commande de Base

1- Générer l'architecture du projet

```
mvn archetype:generate
```

2- Choisir le code par défaut **(1766)**

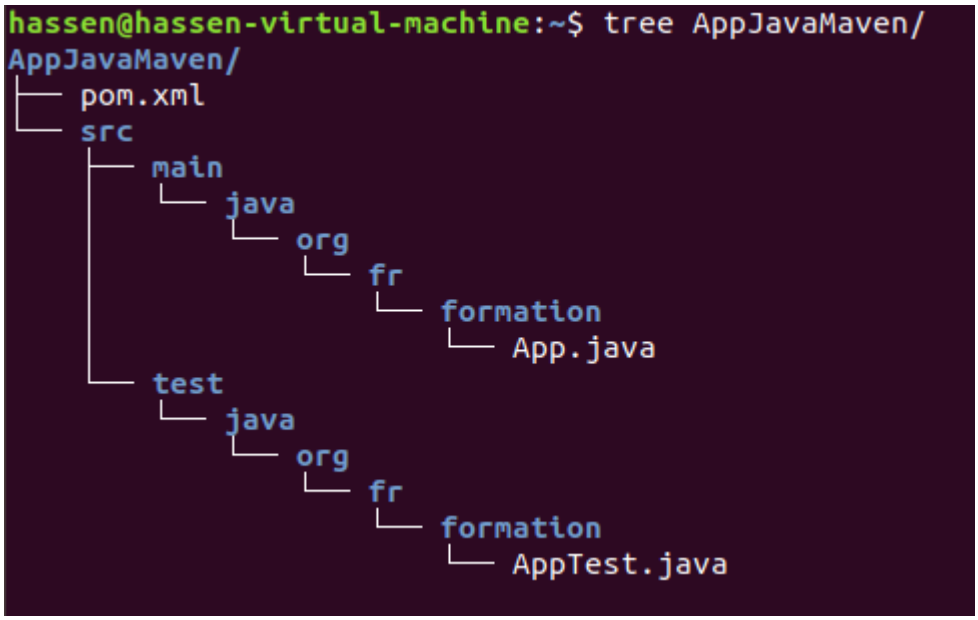
3- Fixer les valeurs des attributs suivants :

- groupId : org.fr.formation
- artifactId : AppJavaMaven
- version: 1.0-SNAPSHOT : (Laisser vide)
- Y : Y

```
Choose a number or apply filter (format: [groupId:]artifactId, case
sensitive contains): 1723:
Choose org.apache.maven.archetypes:maven-archetype-quickstart version:
1: 1.0-alpha-1
2: 1.0-alpha-2
3: 1.0-alpha-3
4: 1.0-alpha-4
5: 1.0
6: 1.1
7: 1.3
8: 1.4
Choose a number: 8:
Define value for property 'groupId': org.gk.cusrsusdevops
Define value for property 'artifactId': AppJavaMaven
Define value for property 'version' 1.0-SNAPSHOT: :
Define value for property 'package' org.gk.cusrsusdevops: :
Confirm properties configuration:
groupId: org.fr.formation
artifactId: AppJavaMaven
version: 1.0-SNAPSHOT
package: org.fr.formation
```

```
Y: : Y
```

4- Vérifier l'architecture du projet



5- Compiler le projet

```
cd AppJavaMaven
mvn compile
```

6- Créer un package

```
mvn package
```

7- Vérifier le résultat

```
tree
```

8- Tester le projet

```
java -cp ./target/AppJavaMaven-1.0-SNAPSHOT.jar org.fr.formation.App
```

9- Créer la documentation de l'application

```
mvn site
```

10- Nettoyer le projet

```
mvn clean
```

11- Testez la construction du projet AppJavaMaven en exécutant la commande

```
mvn clean install
```

Partie B :

Gérer les dépendances d'un projet automatiquement avec Maven

1- Modifier le code source de classe java principale App.java

```
package org.gk.cusrsusdevops;
import org.slf4j.*;
public class App{
    public static void main( String[] args ){
        System.out.println( "Hello World!" );
        Logger  logger = LoggerFactory.getLogger(App.class);
        logger.info("Hello world");
    }
}
```

2- Compiler le projet et analyser le résultat

3- Ajouter les dépendances du package slf4j (https://mvnrepository.com/)

```
...
    <dependency>
      <groupId>org.slf4j</groupId>
      <artifactId>slf4j-api</artifactId>
      <version>1.7.30</version>
    </dependency>
...
```

4- Compiler à nouveau le projet et analyser le résultat

Partie C :

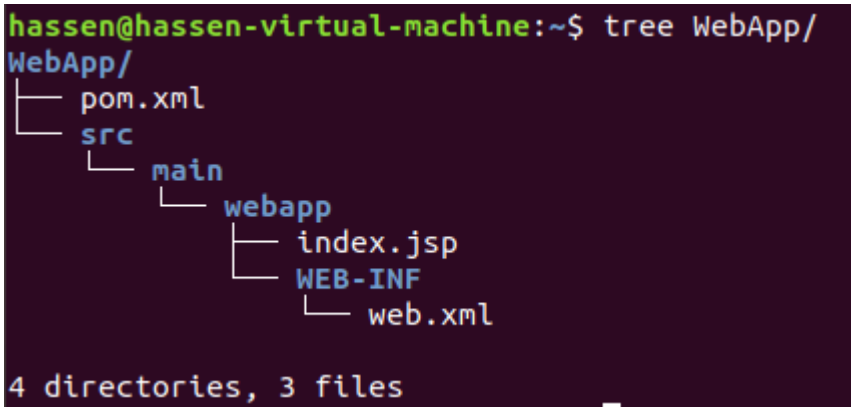
Générer une Application Web

1- Générer l'architecture du projet WebApp (1771: An archetype which contains a sample Maven Webapp project)

```
mvn archetype:generate
```

2- Vérifier l'architecture du projet

```
tree WebApp
```



3- Ajout du plugin Jetty pour tester rapidement l'application web créée. Modiifier le fichier pom.xml et ajouter la définition du plugin

```
...
<plugin>
    <groupId>org.mortbay.jetty</groupId>
    <artifactId>maven-jetty-plugin</artifactId>
    <version>6.1.26</version>
</plugin>
...
```

4- Exécuter le plugin Jetty

```
mvn jetty:run
```

5- Tester l'application à partir d'un navigateur web sur le port 8080

6- Modifier le code de la page index.jsp et vérifier les mises à jour